SNMP와 이동에이전트의 해석적 모델 및 성능 평가

정회원 이 정 우, 윤 완 오, 신 광 식, 최 상 방*

Analytical Models and Performance Evaluations of SNMP and Mobile Agent

Jung-Woo Lee, Wan-Oh Yoon, Kwang-Sik Shin, Sang-Bang Choi* *Regular Members*

요 약

최근 인터넷, 인트라넷 등과 같은 네트워크의 급속한 발전에 따라 많은 네트워크 구성요소를 체계적으로 관리할 필요성이 커지고 있다. 네트워크 규모의 급속한 성장은 기존의 SNMP(Simple Network Management Protocol), CMIP(Common Management Information Protocol) 등을 기반으로 한 클라이언트-서버(client-server) 관리 패러다임으로는 한계를 가진다. 따라서 네트워크를 효율적으로 관리하기 위해서 최근 분산형(distributed) 패러다임인 이동에이전트(Mobile Agent)를 네트워크 관리에 이용하려는 연구가 많이 이루어지고 있다. 본 논문에서는 중앙 집중형의 SNMP, 분산형의 이동에이전트, 그리고 이들 두 접근 방법의 단점을 극복하기 위한 이동에이전트의 한 형태인 혼합모드의 해석적 모델을 제안하고 그 성능을 비교 분석한다. 제안한 해석적 모델을 네트워크 응답 시간에 중점을 두어 성능 평가 한 결과 LAN에서는 대체적으로 SNMP가 유리한 반면 WAN에서는 네트워크 환경에 따라 이동에이전트 또는 혼합모드가 더 좋은 응답 시간을 보임을 알 수 있다. 또한 해석적 방법의 결과를 바탕으로 네트워크 환경, 지연(delay), 태스크(task), 관리 노드 수를 고려한 적응형 네트워크 관리 알고리즘을 제안한 후 실험하였다. 그 결과 적응형 네트워크 관리 알고리즘을 사용하였을 때, 이동에이전트 또는 혼합모드 네트워크 관리 패러다임을 사용하는 것과 비교할 때 약 10%의 성능향상이 있음을 알 수 있다.

Key Words : SNMP, mobile agent, NMS, adaptive network management

ABSTRACT

As the public Internet and private intranet have grown from small networks into large infrastructures, the need to more systematically manage the large number of network components within these networks has grown more important as well. The rapid growth of network size has brought into question the scalability of the existing centralized model, such as SNMP(Simple Network Management Protocol) and CMIP(Common Management Information Protocol). Thus, for efficient network management, researches about mobile agent have also been performed recently. This paper presents analytical models of centralized approach based on SNMP protocol, distributed approach based on mobile agent, and mixed mode to make up for shortcommings of SNMP and mobile agent. We compare the performance of these analytical models based on network management response time. Experiment results show that performance of mobile agent and the mixed mode is less sensitive to the delay in WAN network environment. However, SNMP is more efficient for the simple network environment like LAN. We also propose an adaptive network management algorithm in consideration of network environment, delay, task, and the number of nodes based on the results of analytical models. The results show that the adaptive network management algorithm can reduce the network management response time by 10% compared with either mobile agent or mixed mode network management algorithm.

* 인하대학교 전자공학과 컴퓨터구조및 네트워크연구실(sangbang@inha.ac.kr)
논문번호 : 030009-0107 , 접수일자 : 2003년 1월 7일
※본 과제(결과물)는 정보통신부의 정보통신기초기술연구지원사업(정보통신연구진흥원)으로 수행한 연구결과입니다.(C1-2003-2000-0198)

Ⅰ. 서 론

일반적인 네트워크 관리 시스템은 하나 혹은 여러 개의 NMS(Network Management Station)와 특정한 프로토콜을 이용해 통신할 수 있는 네트워크 내의 에이전트들로 구성된다[1]. 네트워크 NMS와 에이전트들 사이에 특정한 정보를 주고받는 것이 네트워크 관리의 기본이다. 에이전트는 소프트웨어로서 NMS로부터의 요구에 응답하는 기능을 갖고 있다. NMS와 관리 노드(managed node) 사이를 통신하는 정보는 MIB(Management Information Base)에 정의되어 있다[1].

대부분의 네트워크 관리 시스템은 중앙 집중형 방식으로 대표적인 프로토콜인 SNMP(Simple Network Management Protocol), CMIP(Common Management Information Protocol)등을 사용한다. 이들 프로토콜은 NMS가 사용자 인터페이스를 제공하는 클라이언트로서 동작하고 에이전트는 MIB에 저장된 정보에 원격 접근할 수 있도록 서버로서 상호 동작하는 클라이언트-서버(client-server) 수행 방식을 사용한다.

이 중 SNMP는 TCP/IP(Transmission Control Protocol/Internet Protocol)를 기반으로 한 네트워크 관리 프로토콜로서 구조가 단순하며 구현도 용이해 현재 가장 널리 사용되는 프로토콜이다. 그러나 이러한 중앙 집중형 네트워크 관리는 주기적인 폴링(polling)으로 인해 NMS에게 관리 작업이 집중화되므로 NMS의 처리 부하와 네트워크 트래픽이 증가한다는 문제가 있다[2]. 이러한 문제점을 보완하기 위해 최근 들어서는 네트워크 관리를 위해 이동에이전트를 이용하려는 시도가 많이 이루어지고 있다[3,4,5].

이동에이전트는 사용자들의 요구에 맞추어서 에이전트 자신이 데이터가 저장된 장소로 이동하고 지능을 이용해 사용자가 원하는 정보를 선택적으로 얻어올 수 있는 프로그램을 말한다[6]. 네트워크 관리의 경우에는 이동에이전트가 관리되는 네트워크 관리 대상자들로 이동하며 관리 작업을 수행하므로 관리 작업의 분산화와 확장이 용이하며 네트워크 대역폭을 줄일 수 있는 장점이 있다.

이러한 이동에이전트의 이동성이라는 특성은 원격 통신비용을 줄일 수 있다는 장점이 있기는 하지만 노드 방문 시 획득한 데이터들이 계속 누적되기 때문에 오히려 네트워크 응답시간이 증가될 수 있다는 단점도 있다. 따라서 네트워크 관리를 위한 이동에이전트의 정확한 성능을 비교 평가하기 위해서는 직관적인 평가보다 실제 수행 능력을 수치적으로 평가할 수 있는 해석적 성능 평가 모델이 필요하다.

이동에이전트는 데이터를 누적시키며 이동하는 전형적인 구조 외에도 SNMP의 경우 메시지 교환 횟수가 많다는 단점과 이동에이전트의 경우 누적되는 데이터 양이 많다는 단점을 각각 극복하기 위한 혼합모드도 고려한다. 혼합모드는 이동에이전트가 방문한 노드에서 수집한 데이터를 누적시켜 이동하는 것이 아니라 방문한 노드의 임시 보관소에 데이터를 보관하고 다른 노드로 이동하는 방식이다. 이 데이터는 추후 TCP/IP 연결을 통해 클라이언트-서버 방식으로 NMS에게 전달된다.

본 논문에서는 SNMP, 이동에이전트, 그리고 혼합모드를 사용한 네트워크 관리의 성능을 양적으로 평가하기 위한 해석적 모델을 만든 후 이를 바탕으로 네트워크 환경에 알맞은 적응형 네트워크 관리 알고리즘을 제안하고 검증한다. 실제 네트워크 관리 시스템의 성능은 프로세싱 시간, 노드의 처리 부하, 보안 등의 문제를 함께 고려하여 평가해야 하지만 본 논문에서는 네트워크 관리에서 가장 비중이 높은 전체 네트워크 관리 정보의 응답시간만을 고려한다.

본 논문의 구성은 다음과 같다. 제 2장에서는 네트워크 관리 패러다임인 클라이언트-서버 방식의 SNMP와 분산형 방식의 이동에이전트에 대해 알아보고 제 3장에서는 LAN과 WAN 환경에서 두 네트워크 관리 패러다임을 비교하기 위한 해석적 평가 모델을 제시한다. 제 4장에서 실험을 통한 해석적 모델의 검증을 한 후 이를 바탕으로 적응형 네트워크 관리 알고리즘을 제안하고 검증한다. 제 5장에서 향후 연구 과제와 결론을 제시한다.

Ⅱ. 네트워크 관리 패러다임

2.1 SNMP

현재 IP 네트워크 관리를 위해 보편적으로 사용되는 네트워크 관리 프로그램은 IETF(Internet Engineering Task Force)에 의해 제안된 SNMP와 ISO(International Organization for Standardization)에서 제정한 CMIP을 기반으로 한 프로토콜을 사용한다.

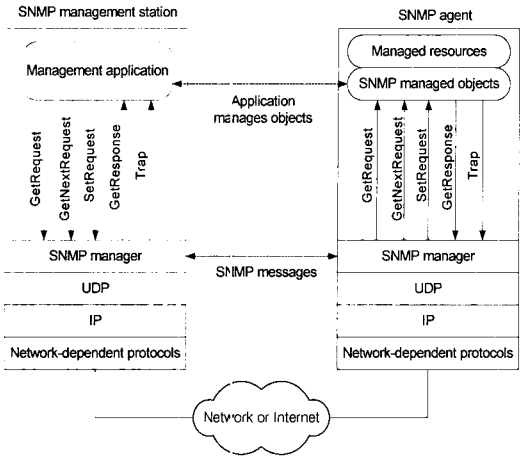


그림 1. SNMP 프로토콜의 동작

이들 프로토콜들은 NMS(클라이언트)가 각 네트워크 장치에 있는 에이전트(서버)들로부터 원하는 정보를 폴링에 의해 얻어와 NMS에서 데이터를 처리하는 클라이언트-서버 패러다임에 기반을 둔다. 각 에이전트는 네트워크 장치에 맞는 적절한 파라미터를 계층적으로 저장한 MIB를 관리하는 역할을 한다. 클라이언트-서버 패러다임에서는 네트워크 관리에 관련된 모든 계산은 NMS에서 수행 되어진다.

관리노드의 SNMP 에이전트는 자신의 MIB를 관리하는 부분과 NMS와 통신하는 부분으로 구성되어 있다. SNMP를 이용하여 관리되는 특정한 정보와 자원 등을 모아 놓은 집합체가 MIB이다. 네트워크를 관리한다는 것은 관리 대상인 장비들이 제공하는 MIB 중에서 특정 값을 얻어와 그 장비의 상태를 파악하거나 그 값을 변경함을 의미한다[7].

NMS와 SNMP 에이전트 간에 통신을 위해 사용되는 프로토콜이 SNMP이며 GetRequest, GetNextRequest, SetRequest, GetResponse, Trap의 5가지 메시지를 주고받는다. NMS는 폴링 주기에 따라 에이전트 정보 요구로 GetRequest 또는 GetNextRequest 메시지를 전송하며 에이전트는 에이전트 정보를 포함한 응답으로 GetResponse 메시지를 전송한다. 이러한 형태의 폴링에서 NMS는 하나의 에이전트에 정보를 요청하고 그에 대한 응답을 기다리는 동안에 다른 에이전트에게는 정보를 요구할 수 없다. 그림 1은 NMS와 에이전트 사이의 SNMP 프로토콜 동작을 보여준다.

클라이언트-서버 구조는 NMS에 과도한 트래픽과 부하를 유발한다. 특히 네트워크가 매우 혼잡한 상태일 때는 매우 비효율적이다. 네트워크 혼잡이 발생한 동안 NMS는 장치들의 구성을 바꾸기 위한 노력을 하는데 이는 장치들과 더 많은 상호 작용을 요구할 것이고 네트워크 관리가 오히려 혼잡을 가중시키는 결과를 초래한다.

이러한 클라이언트-서버 패러다임에서의 문제들을 개선하기 위해 IETF와 ISO는 몇 가지를 제안함으로써 관리 구조를 개선시켜 왔다. SNMPv2에서 프락시 에이전트의 개념을 통해 계층적 분산화를 소개했다[8]. 또한 SNMP로부터 파생된 RMON (Remote Monit -oring)은 probes라 불리는 독립된 장치를 각 LAN 세그먼트 당 설치해 링크 상태, 트래픽 패턴 등의 전체적인 정보를 관리할 수 있도록 했다[9]. 그러나 이들은 최근 네트워크를 관리하기 위해 요구되는 수준의 분산화를 제공하지 못하는 것으로 여러 실험 결과 밝혀졌다[2].

2.2 이동에이전트

2.2.1 이동에이전트의 개요

에이전트라는 용어의 사전적인 의미는 대행자, 대리인 정도의 의미를 가지지만 컴퓨터 분야에서는 작업을 대행해주는 프로그램으로 해석될 수 있다. 학문적인 의미에서 에이전트는 분산 환경에서 작업을 수행하는 지적인 특성을 갖는 응용 프로그램으로 정의한다. 이러한 정의는 각 연구 분야에 따라서 다양한 의미로 해석이 가능해진다.

이동에이전트는 프로그램 자체가 네트워크나 인터넷을 돌아다니며 수행되는 프로그램으로서 한 호스트로 옮겨간 뒤에 원하는 동작을 수행하며 목적을 위해 필요에 따라 다른 이동에이전트들과 통신을 할 수도 있다. 자바 애플릿의 경우 서버에서 브라우저로 이동하는 형태는 이동에이전트와 유사하지만 애플릿은 브라우저의 요구에 의해서 이동하는 형태고 이동에이전트는 자신의 판단에 의해 이동한다는 점이 다르다. 즉, 이동에이전트는 자율성이 있다는 것을 의미한다.

이동에이전트의 대표적 응용분야는 전자상거래이다. 예를 들어 사용자가 원하는 제품을 에이전트가 여러 쇼핑몰을 돌아다니면서 가장 조건이 좋은 서버에서 거래를 할 수 있도록 한다. 정보 검색 측면에서 보면 여러 데이터베이스 서버나 검색엔진 서버들 사이를 움직이면서 사용자가 원하는 정보만을 찾아서 제공할 수 있다. 그밖에도 이동에이전트는

네트워크 관리에도 사용될 수 있다. 에이전트가 네트워크를 이동하면서 트래픽이 많은 곳을 찾아서 문제를 해결하거나 NMS에게 문제점을 알려줌으로서 네트워크 문제를 해결할 수 있다.

이러한 기능을 수행하기 위해서 이동에이전트는 자신의 판단에 의해서 이동하는 기능뿐만 아니라 동일한 이동에이전트를 복제해서 다른 호스트 서버로 보내고, 다른 이동에이전트와 통신을 해서 원하는 정보의 위치를 파악하는 기능을 수행한다. 이렇게 수행된 여러 가지 결과를 종합하여 최종결과를 만들어 낸다[10].

2.2.2 네트워크 관리를 위한 이동에이전트

기존의 클라이언트-서버 방식의 네트워크 관리는 NMS에 관리 작업이 집중화되므로 NMS의 처리 부하와 네트워크 트래픽이 증가한다는 문제가 있다. 이러한 문제점을 보완하기 위해 최근 들어서는 이동에이전트를 이용한 네트워크 관리의 분산화 연구가 진행되고 있다.

이동에이전트는 사용자가 요구한 작업을 현재의 NMS에서 수행하지 않고 실제 그 작업을 처리하는 관리 노드로 이동시켜 수행함으로써 수행의 효율을 높이고 네트워크 부하를 줄이는 효과를 유도한다. 즉, 기존의 클라이언트-서버 개념과 달리 서버의 내용을 클라이언트를 통해 전송 받아 정보를 얻거나 작업을 수행하는 것이 아니라 클라이언트가 필요로 하는 작업을 위해 이동에이전트를 서버로 보내어 수행하는 것이다.

그림 2는 네트워크 관리에서 SNMP를 사용한 클라이언트-서버 수행 방식과 이동에이전트의 수행 방식을 도식화하여 비교한 것이다. SNMP의 수행 방식은 주기적 혹은 필요에 따라 NMS에서 SNMP에이전트로 메시지 요구를 보낸 후 응답을 받는 구조로 한 번에 한 개씩의 요구를 처리하지만 이동에이전트는 네트워크 관리를 위해 NMS를 떠나면 NMS와 비동기적으로 관리를 수행하고 사전에 약속된 관리 노드를 순회한 후 NMS로 돌아오는 수행 방식을 갖는다.

이동에이전트는 관리 효율을 향상시키고 프로세싱과 제어를 분산화 할 수 있다. 네트워크 관리 측면에서 이동에이전트의 활용은 비용의 감소, 비동기적인 프로세싱, 분산된 프로세싱, 유연성을 제공한다[11]. 그러나 이동에이전트를 사용하는 경우 관리되는 노드들에서 획득된 데이터가 이동에이전트에 누적됨에 따라 노드 간을 이주하는데 전송 시간이

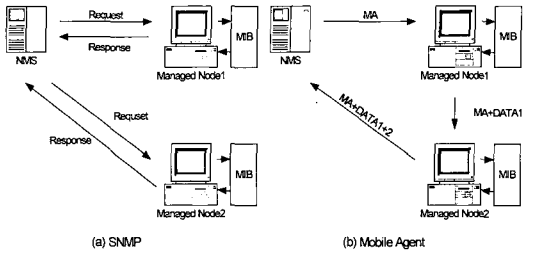


그림 2. SNMP와 이동에이전트의 수행 방식 비교

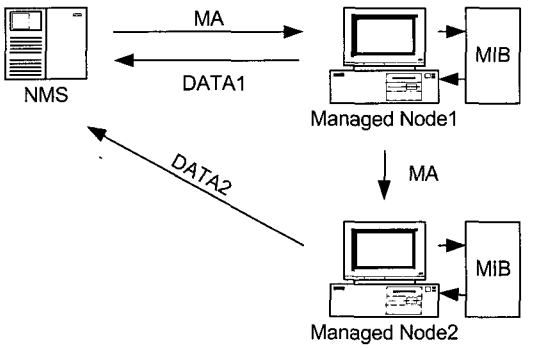


그림 3. 혼합모드의 수행 방식

길어지는 단점이 있다[2]. SNMP를 사용하는 경우 NMS와 관리 노드들 간에 메시지 교환 횟수가 많다는 단점과 이동에이전트를 사용한 경우 노드 간 이주시 관리 데이터의 누적으로 인해 전송 지연이 발생하는 단점을 극복하기 위해 두 방식을 혼합해서 사용하기도 한다[4,12]. 즉, 이동에이전트가 방문한 관리 노드에서 수집한 데이터를 누적시켜 이동하지 않고 관리 노드의 임시 보관소(locker)에 저장해 두었다가 그 데이터를 이동에이전트와 상관없이 직접 NMS로 보내고 이동에이전트는 관리하고자 하는 다음 에이전트로 이동한다. 또한 보관소에 따로 저장된 데이터는 암호화 후 보낼 수 있어 보안에 유리하고 네트워크 트래픽의 감소 및 데이터 무결성을 향상시키는 장점을 갖는다.

본 논문에서는 이와 같은 네트워크 관리 방법을 혼합모드(mixed mode)라 부르며, 그림 3은 그 수행 방식을 도식화 한 것이다. NMS에서 네트워크 관리를 위해 떠난 이동에이전트는 첫 번째 에이전트를 방문한 후 관리 데이터를 이동에이전트에 누적시키지 않고 관리노드에 데이터를 임시 저장한 후 NMS로 TCP/IP를 사용하여 보내준다.

다음 장에서는 네트워크 관리 시스템의 성능 척도로 SNMP, 이동에이전트, 혼합모드의 각 수행 방

식에 따른 전체 네트워크 관리 정보의 응답 시간을 구하기 위한 해석적 모델을 제시한다.

Ⅲ. 해석적 모델

3.1 해석적 모델을 위한 네트워크 환경

본 논문에서는 관리되는 네트워크 상의 NMS의 위치에 따라 네트워크 환경을 다음과 같이 세 가지로 분류하여 해석적 모델의 성능을 평가한다.

(1) NMS가 어느 한 네트워크에 속하고 동일한 한 개의 네트워크를 관리하고자 하는 경우 - LAN환경

(2) NMS가 어느 한 네트워크에 속하고 동일한 여러 개의 네트워크를 관리하고자 하는 경우 - 인트라넷(intranet) 환경

(3) NMS가 어느 한 네트워크에 속하고 지역적으로 먼 다른 네트워크를 관리하고자 하는 경우 - 인터넷(internet) 환경

본 논문에서는 위 (1)에서 제시한 이더넷(ethernet) LAN으로 구성된 하나의 네트워크를 충돌 도메인(collision domain)이라 부르며, (2)와 (3)에서 제시한 인트라넷과 인터넷 환경을 통합적으로 WAN이라 한다. LAN 환경은 소규모의 네트워크 관리를 위해 사용되고, 인트라넷 환경은 학교, 기업 등에서 네트워크를 관리하고자 할 때 적용할 수 있다. 또한 인터넷 환경은 회사 단위 또는 지역적으로 떨어져 있는 곳을 원격으로 관리하고자 하는 경우 적용할 수 있다.

3.2 LAN환경에서 패킷 전달시간

LAN은 이더넷으로 구성되며 전송 프로토콜로서 IEEE 802.3 표준에 규격화되어 있는 CSMA/CD(Carrier Sense Multiple Access/ Collision Detection)를 사용한다. 이더넷은 데이터를 송신하려는 클라이언트가 네트워크 상에 다른 컴퓨터가 통신하고 있는지를 조사해 신호가 송출되고 있지 않을 시 데이터를 전송하는 구조다. 동시에 여러 노드에서 데이터를 전송할 경우 충돌이 발생한다. CSMA/CD는 이 충돌을 감시하는데, 충돌이 발생한 경우에는 일정 시간 갖고 있다가 다시 신호를 보내 통신을 제어한다. 그림 4와 같이 각 노드는 CSMA/CD프로토콜을 사용하는 이더넷으로 구성된 네트워크가 있을 경우 가장 멀리 떨어져 있는 노드 A와 B를 고려했을 때 버스를 통해 메시지가 성공

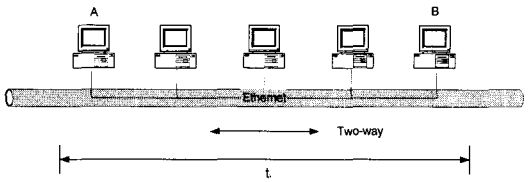


그림 4. 이더넷 네트워크

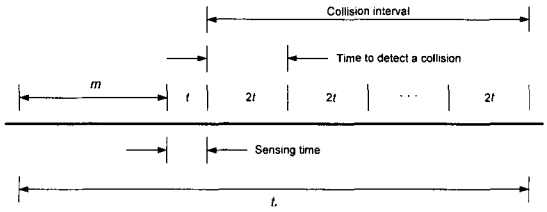


그림 5. CSMA/CD에서 가상 응답시간의 계산

적으로 도달하는 평균 시간을 구할 수 있다. 노드 A와 B에 초점을 맞춘다면 메시지를 전송하기 위해 요구되는 최대 시간을 구할 수 있다. 메시지를 성공적으로 전송하는 걸리는 시간을 가상 응답시간(virtual transmission time) t_v라고 부른다. t는 A에서 B로 메시지를 보내는 전달 지연(propagation delay)이다[13].

가상 응답시간은 세 가지 구성 요소를 가진다. 메시지를 전송하기 위해 요구되는 시간 m, 메시지 전송 완료를 감지하는 위한 시간 t, 충돌이 발생했을 때 충돌을 해결하기 위해 요구되는 시간 $2tJ$의 합으로 구성된다[13].

$$t_v = m + t + 2tJ = m[1 + a(1 + 2J)] \quad (1)$$

여기서 J는 충돌이 발생했을 경우 메시지의 평균 재전송 횟수를 의미하고 이는 재 전송 방법에 따라 결정되며, $a = t/m$이고 메시지 응답시간 m과 전달 지연 t의 비율을 의미한다. 그림 5는 식 (1)의 가상 응답시간의 의미를 그림으로 보여준 것이다.

CSMA/CD 방법의 성능을 향상시키는 유일한 방법은 파라미터 $a = t/m$을 줄이는 것이다. 예를 들어 10Mbps의 전송율을 가진 네트워크의 경우 보다 짧은 전송 매체의 길이를 사용하거나 프레임 길이를 크게 해서 보내는 것을 의미한다. 파라미터 값 a을 감소시킴에 따라 충돌은 보다 쉽게 감지되고 시스템은 그림 6과 같이 중앙 집중된 M/G/1 큐의 특

성을 보인다. 한 충돌 도메인을 한 개의 가상 큐로

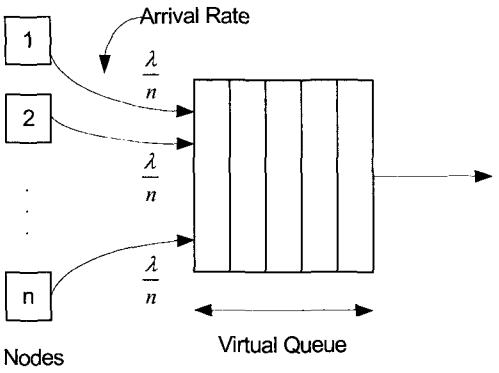


그림 6. 한 충돌 도메인의 큐잉 모델

보고 각각의 노드에서 보내는 메시지들은 충돌이 발생하지 않으면 그대로 전송되지만 네트워크 내에서 충돌이 발생한다면 가상 큐에 저장되었다가 FIFO(first in-first out)방법으로 전송되는 것이다. 가상 큐에서 메시지의 도착율(arrival rate)은 Poisson 특성을 가지며 가상 큐에서의 서비스 시간은 일반 분포를 갖으며 서버는 한 개로 본 것이다.

CSMA/CD를 사용하는 네트워크 내에서 한 프레임을 전달하기 위한 시간은 식 (2)과 같다. 식 (2)은 최고의 성능과 수식의 간략화를 위해 $a=0$으로 했을 때 한 노드로부터 다른 노드로 데이터를 전달하기 위해 필요한 평균 전달 시간(average transfer time) t_f을 나타낸다. 식 (2)은 일반적인 M/G/1 큐에서 대기시간(wait time)과 응답시간을 더한 평균 지연(average delay)을 구하는 형태와 같음을 보여준다[14].

$$t_f = m + \frac{\lambda \overline{m^2}}{2(1-\rho)} \quad (2)$$

여기서 m은 평균 프레임 전송시간, λ은 총 평균 트래픽(frames/sec)으로 단위 시간에 각 노드에서 보내고자 하는 전체 프레임의 양으로 $\rho = \lambda m$이며, 충돌 도메인 내의 네트워크 트래픽의 정도를 의미한다. 또한 m은 다음 식을 사용하여 구할 수 있다.

$$m = [E(L_p) + L_h]/C \quad (3)$$

여기서 $E(L_p)$는 평균 패킷 크기, L_h는 헤더 크기, C는 전송 매체의 전송율을 의미한다.

본 논문에서는 해석과 계산을 용이하게 하기위해 프레임 전송시간 m은 일정한 상수 값을 갖는다고 본다. m을 일정한 상수 값을 갖게 하기 위해 $[E(L_p) + L_h]$을 이더넷의 MTU(Maximum Transmission Unit)로 대체한다. 따라서 가상 큐는 Poisson 도착율을 가지고 프레임 길이 분포는 일정하므로 M/D/1 큐와 같이 동작하다고 볼 수 있다. M/D/1 큐는 식 (2)에서 $\overline{m^2} = m^2$이 되고 $\rho = \lambda m$이므로 평균 전달 지연 값 t_f는 식 (4)과 같이 유도된다. 평균 전달 시간 t_f는 본 논문에서 LAN환경에서(한 충돌 도메인 내에서) 노드간의 패킷 전달시간으로 사용된다.

$$t_f = \frac{m}{(1-\rho)} \times (1 - \frac{\rho}{2}) \quad (4)$$

3.3 LAN환경을 고려한 해석적 모델

본 절에서는 네트워크 관리 패러다임인 SNMP, 이동에이전트, 혼합모드에 대한 각각의 해석적 모델을 식 (4)에서 구한 평균 전달시간을 이용해서 네트워크 응답 시간을 구해본다. 실제 네트워크 관리 시스템의 성능은 프로세스의 처리 시간, 노드의 처리 부하, 보안 등의 문제를 고려하여 평가해야 하지만 본 논문에서는 전체 네트워크 관리정보의 네트워크 응답시간만을 고려한다.

3.3.1 SNMP

SNMP를 사용하는 경우 네트워크 관리에서 가장 큰 비중을 차지하는 작업이 GetRequest와 GetResponse를 처리하는 부분이기 때문에 이 부분에 주안점을 두어 모델링 한다. NMS는 관리 노드의 상태 파악을 위해 필요한 GetRequest 메시지를 보내고 관리 노드는 요구 받은 SNMP 변수의 값(MIB 객체)을 NMS에게 GetResponse 메시지를 사용해서 보내는 클라이언트-서버 방식이다. 네트워크 관리 태스크에 따라 SNMP는 한 개의 MIB 객체만 필요한 경우가 있는 반면에 interface utilization과 같은 작업은 한 관리 노드 당 세 개의 MIB 객체를 필요로 한다. 관리를 위해 이러한 상태 함수(health check activity)를 필요로 하는 경우 한 개의 노드를 여러 번 방문해야 하는 경우가 있다[15].

그림 7은 SNMP 메시지 포맷을 보여준다. SNMP는 UDP(User Datagram Protocol)과 IP 헤더를 붙

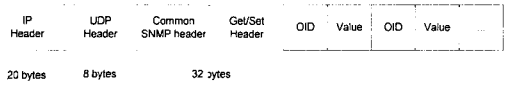


그림 7. SNMP 메시지 포맷

여서 사용하고 MIB 객체의 OID(Object Identifier)을 요구 메시지에 보내고 응답메시지로 그에 대한 값을 반환 받는다. 일반적 SNMP 메시지는 작기 때문에 이더넷의 MTU 크기를 넘어서지 않는다. 그러므로 SNMP메시지는 일반적으로 요구와 응답 모두에서 패킷 분할(fragmentation)이 발생하지 않는다.

SNMP를 사용했을 경우 전체 네트워크 응답시간 T_{SNMP}는 식 (5)과 같이 표현된다.

$$T_{SNMP} = 2\sum_{n=1}^{N}\sum_{r=1}^{R} t_f \qquad (5)$$

여기서 N은 관리하고자 하는 노드 수, R은 MIB 객체를 얻기 위해 동일한 관리 노드를 방문하는 횟수, t_f는 식 (4)에서 얻은 충돌 도메인 내에서의 노드간의 패킷 전달시간을 각각 나타낸다. 즉, SNMP로 네트워크를 관리하면 각 관리되는 노드에게 한 번씩 혹은 R번의 요구를 보낸 후 이에 대한 응답을 받아야 함으로 $2t_f \times N \times R$의 시간의 소요된다.

3.3.2 이동에이전트

이동에이전트는 NMS로부터 관리하고자 하는 첫 번째 노드로 이동한 후 NMS와 비동기적으로 검색한 MIB 객체 값을 이동에이전트의 데이터 영역에 계속 누적시키면서 각각의 관리 노드들을 이동하는 분산형 관리를 수행한다.

이동에이전트의 자체 크기는 이동에이전트의 코드의 크기 M_{code}, 상태 정보의 양 M_{state}, 누적되는 데이터의 양 M_{data}의 합인 M으로 표현된다. 이동에이전트의 전송을 위해서 M에 ATP(Agent Transfer Protocol) 헤더가 붙고 ATP헤더가 붙은 M_A는 네트워크에서 전송을 위해 TCP/IP 헤더가 추가된 후 전송되는 최종 이동에이전트의 크기는 M_s가 된다[2]. 식 (6)은 위의 과정을 표현한 것이다.

$$M = M_{code} + M_{state} + M_{data}$$
$$M_A = ATP\_header + M$$
$$M_s = (TCP\_header + IP\_header) + M_A \qquad (6)$$

보통 이동에이전트의 크기는 이더넷의 MTU 크기보다 크므로 패킷 분할이 발생하고 분할된 패킷은 각각의 노드를 방문 할 때 재조립(reassembly) 후 원하는 MIB 객체들을 얻는다. 이 때 MIB 객체 값들은 이동에이전트의 데이터 영역에 계속 누적시키면서 이동한다. 그러므로 누적된 데이터 영역이 MTU 크기를 넘어서면 추가의 패킷이 더 전송된다. 따라서 전체 네트워크 응답시간 T_{MA}는 식 (7)과 같이 나타낼 수 있다.

$$\begin{aligned} T_{MA} = & \Big[\sum_{i=1}^{I} N_{MA} + \sum_{i=I+1}^{2I}(N_{MA}+1) \\ & + \sum_{i=2I+1}^{3I}(N_{MA}+2) + \cdots \\ & + \sum_{i=(k-1)I+1}^{kI}(N_{MA}+(k-1)) \\ & + \sum_{i=kI+1}^{N}(N_{MA}+k) \\ & + (N_{MA}+k)\Big] \times t_f \end{aligned} \qquad (7)$$

$$= \Big[\sum_{k=0}^{N/I}\sum_{i=kI+1}^{N}(N_{MA}+k) + (N_{MA}+k)\Big] \times t_f$$

여기서 $N_{MA} = M_s / MTU$, $I = MTU / MIB_{value}$ 이다. N_{MA} 위 식에서 M_s는 실제 이동에이전트의 크기, 는 이동에이전트를 한번에 전송 가능한 MTU 크기만큼씩 패킷 분할한 값으로 노드간의 이주시 이동에이전트가 전송되는 패킷의 개수, MIB_{value}는 한 개의 관리 노드를 방문할 때 이동에이전트에 추가되는 MIB 객체 크기들의 합, I는 MTU 크기를 MIB_{value} 크기로 나눈 값으로 이동에이전트가 관리 노드들을 방문하면서 누적된 값들이 MTU 크기를 넘어서 추가로 패킷을 더 보내야 되는 관리 노드 개수의 임계 값을 의미한다.

그러므로 식 (7)은 관리를 위해 NMS를 떠난 이동에이전트는 패킷 분할이 발생하고 I만큼씩의 관리 노드들을 방문 할 때마다 MIB_{value}값의 누적으로 인해 추가로 패킷이 증가하며 관리 노드 수 N에 따라 k개 까지 패킷이 증가할 수 있음을 보여준다. 여기서 이동에이전트는 NMS에서 미리 정해진 관리 작업을 할당 받기 때문에 각 관리 노드에서 얻는 MIB_{value} 크기는 일정하다고 가정한 것이다.

3.3.3 혼합모드

혼합모드에서의 이동에이전트는 각 관리 노드를 순회하면서 원하는 작업을 수행하고 각각의 노드를 방문할 때 획득한 MIB 객체 값은 관리 노드에 임시로 저장된다. 이동에이전트가 모든 노드에서 관리 작업을 수행하면 각 노드에 저장된 MIB 객체 값들은 NMS로 TCP/IP를 사용해 순차적으로 전송된다. 혼합모드의 전체 네트워크 응답 시간 T_{Mixed}은 식 (8)과 같이 표현된다.

$$T_{Mixed} = \sum_{n=1}^{N} N_{MA} \times t_f + \sum_{n=1}^{N} t_f = \sum_{n=1}^{N} (N_{MA}+1) \times t_f \quad (8)$$

혼합모드에서는 이동에이전트와 다르게 추가의 데이터 누적이 없기 때문에 추가의 패킷 증가를 고려할 필요가 없지만 관리 노드에 저장된 데이터를 TCP/IP로 NMS에게 실어 나르기 위한 추가의 응답 시간이 필요한 것을 알 수 있다.

3.4 WAN환경을 고려한 해석적 모델

WAN환경에서는 관리 노드들이 속한 여러 충돌 도메인이 존재할 수 있으며 충돌 도메인 간에 대역폭(bandwidth)과 지연(delay)을 고려해야 한다.

그림 8은 WAN환경을 고려한 네트워크 토폴로지(topology)의 예를 보여준다. 인트라넷과 인터넷 환경을 고려하기 위해 관리되는 충돌 도메인 사이에 경로 AS(transit autonomous system)을 두어서 패킷 전송에 걸리는 지연 시간을 적용한다[16]. NMS는 관리 도메인(management domain)에 위치하고 관리 도메인에는 관리 노드가 존재하지 않으며 관리 노드들이 있는 충돌 도메인 개수는 c개까지 있다.

SNMP를 사용한 경우 각 요구 메시지 당 응답 메시지가 모두 경로 AS을 거쳐서 관리가 이루어진다. 즉, NMS를 떠난 요구 메시지는 관리 도메인, 경로 AS, 관리 노드가 있는 충돌 도메인을 거친 후 응답 메시지는 역순의 과정으로 돌아온다. 식 (9)은 전체 네트워크 응답 시간 T^{W}_{SNMP}을 보여준다.

$$T^{W}_{SNMP} = T_{SNMP\_M} + T_{SNMP\_1} + T_{SNMP\_2} + \cdots + T_{SNMP\_c} + (N \times 2D) \quad (9)$$

여기서 N은 충돌 도메인에 속한 모든 관리 노드의 수이고 D는 경로 AS에서의 패킷 지연이다. 전체 네트워크 응답시간은 관리 도메인에서 T_{SNMP_M}만큼, 충돌 도메인 c개에서 총 $T_{SNMP_1} + T_{SNMP_2} + \cdots + T_{SNMP_c}$만큼 그리고 경로 AS에서는 $N \times 2D$만큼의 합으로 표현된다. 또한, T_{SNMP_M}와 T_{SNMP_1}, T_{SNMP_2}, …, T_{SNMP_c} 각각은 식 (5)에서 구한 T_{SNMP}을 이용해서 구할 수 있다. 식 (9)에서 보듯이

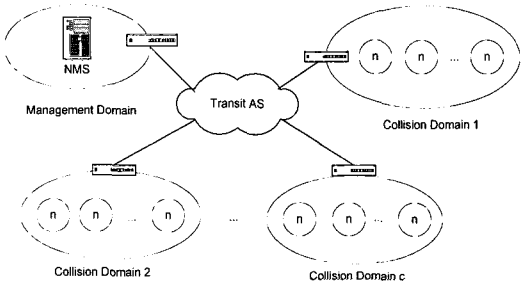


그림 8. WAN 환경을 고려한 네트워크 토폴로지

SNMP는 경로 AS의 지연에 민감하다는 것을 알 수 있다. 이는 모든 관리 작업이 경로 AS을 거쳐서 이루어지기 때문이다.

WAN에서 이동에이전트를 사용한 네트워크 관리 작업의 해석적 모델을 얻기 위해 NMS는 관리 도메인에 위치하고 한 개의 충돌 도메인에 있는 관리 노드들만 관리하는 경우를 먼저 고려해 본다. 이 경우 전체 네트워크 응답시간 $T^{W}_{MA_1}$은 다음과 같이 얻어진다.

$$T^{W}_{MA\_1} = N_{MA}(t_f + D) + T_{MA\_1} + (N_{MA} + k)(t_f + D) \quad (10)$$

이동에이전트가 NMS에서 관리하기 위한 충돌 도메인까지 가는 전송 시간은 $N_{MA}(t_f + D)$이고 충돌 도메인 안에서의 응답시간 T_{MA_1}은 LAN환경에서 제안한 수식 (7)으로부터 구할 수 있다. 이동에이전트가 관리 노드들을 모두 순회한 후 다시 NMS로 돌아올 때는 T_{MA_1}에서 얻은 추가된 패킷 k을 고려해서 $(N_{MA} + k)(t_f + D)$만큼의 전송 시간이 필요하다.

수식 (10)을 두 개의 충돌 도메인이 존재하는 경우로 확장하면 전체 네트워크 응답 시간 $T^{W}_{MA_2}$은 다음과 같다.

$$T^{W}_{MA\_2} = N_{MA}(t_f + D) + T_{MA\_1} + (N_{MA} + k_1)(t_f + D) + T_{MA\_2} + (N_{MA} + k_2)(t_f + D)$$

(11)

위 식에서 $(N_{MA} + k_1)(t_f + D)$은 이동에이전트가 첫 번째 충돌 도메인에서 관리 작업을 마친 후 두 번째 충돌 도메인으로 이동하면서 경로 AS을 거쳐 갈 때의 응답시간이다. 이 때 k_1은 T_{MA_1}을 구하는 과정에서 얻어진 추가의 패킷 수를 의미한다. 마찬가지로 두 번째 충돌 도메인에서 T_{MA_2}을 구하고 관리 도메인으로 보내 질 때 응답시간은 $(N_{MA} + k_2)(t_f + D)$가 된다. $T^{W}_{MA_1}$과 $T^{W}_{MA_2}$을 구하고자 할 때 관리 노드의 총 개수가 N개로 같다면 식 (10)에서 얻은 k값과 식 (11)에서 얻은 k_2은 같을 것이다.

관리하고자 하는 네트워크의 충돌 도메인이 c개 만큼 있다면 T^{W}_{MA}로 이동에이전트의 전체 네트워크 응답 시간을 다음과 같이 일반화시킬 수 있다.

$$T^{W}_{MA} = N_{MA}(t_f + D) + \sum_{i=1}^{c} T_{MA\_i} + \sum_{i=1}^{c}(N_{MA} + k_i)(t_f + D)$$

(12)

SNMP의 경우 경로 AS에서의 지연 D는 관리 노드의 총 개수에 의존했으나 이동에이전트는 충돌 도메인 개수와 이동에이전트의 크기에 의존함을 알 수 있다. 이는 이동 에이전트가 경로 AS의 지연에 민감하지 않다는 것을 보여준다.

혼합모드의 네트워크 응답시간을 알아보기 위해 먼저 두 개의 충돌 도메인과 한 개의 관리 도메인에 NMS가 있는 경우를 고려하면 $T^{W}_{Mixed_2}$은 다음과 같다.

$$T^{W}_{Mixed\_2} = N_{MA}(t_f + D) + T_{Mixed\_1} + N_{MA}(t_f + D) + T_{Mixed\_2} + \sum_{n=1}^{N}(t_f + D) \quad (13)$$

위 식에서 두개의 $N_{MA}(t_f + D)$은 이동에이전트가 각각 NMS에서 첫 번째 충돌 도메인으로 이동 할 때와 첫 번째 충돌 도메인에서 두 번째 충돌 도메인으로 이동 할 경우의 전송시간을 나타낸다. 각 충돌 도메인에서는 식 (8)에서 구한 T_{Mixed}을 이용해 응답시간을 구할 수 있으며 각 관리 노드에서 획득한 데이터를 TCP/IP를 사용해 NMS로 보낼 때는 각 관리 노드 당 $(t_f + D)$만큼의 전송시간을 갖는다. 이는 각 노드로부터 메시지가 경로 AS와 관리 도메인을 거쳐서 NMS로 전달되기 때문이다. 식 (13)을 충돌 도메인이 c개만큼 있는 네트워크로 일반화하면 T^{W}_{Mixed}은 다음과 같이 표현된다.

$$T^{W}_{Mixed} = \sum_{i=1}^{c} N_{MA}(t_f + D) + \sum_{i=1}^{c} T_{Mixed\_i} + \sum_{n=1}^{N}(t_f + D) \quad (14)$$

혼합모드에서는 경로 AS에서의 지연은 전체 네트워크 관리 노드수와 충돌 도메인 개수에 의존함을 알 수 있다. 이는 SNMP에 비해 지연에 덜 민감하지만 이동에이전트보다는 지연에 민감한 것이다. 또한 혼합모드는 노드간의 이주시 추가의 패킷 증가가 없으므로 관리 태스크에 상관없이 일정한 응답시간을 가짐을 알 수 있다.

Ⅳ. 해석적 모델의 성능평가

4.1 LAN환경에서의 실험

LAN환경에서 제안한 SNMP, 이동에이전트, 혼합모드의 해석적 모델에 샘플 데이터를 적용하여 실험한다. LAN환경은 한 충돌 도메인 내에 NMS와 관리 노드들이 모두 존재하는 경우이다. 실험에 사용된 공통된 파라미터는 표 1과 같다.

표 1. 공통적으로 사용된 파라미터 값

Parameter	Value
N	248
C	10Mbps
MTU	1500Byte
ρ	0.5
m	$(MTU \times 8)/C$

관리 노드의 총 개수는 248개, 이더넷의 전송율은 10Mbps, 하드웨어에 의해 결정되는 MTU는 이더넷에서 일반적으로 1500byte를 사용하므로 SNMP는 패킷 분할이 없다고 본다. 또한, 충돌 도메인 내의 노드 간 패킷 전송시간에 영향을 미치는 파라미터인 m은 이더넷의 MTU를 전송율로 나눈 값이고 네트워크 내의 트래픽 정도를 나타내는 ρ를 50%로 정한다. 본 논문에서는 응용 프로그램에서의 프

로세싱 시간은 고려하지 않고 네트워크 내의 링크들은 에러가 없다고 본다.

그림 9는 한 노드에서 얻어지는 MIB_{value}는 40바이트, 한 노드로의 요구 회수 R은 1회, 이동에이전트의 크기는 4.5Kbyte인 경우의 관리 노드 수의 증가에 따른 전체 네트워크 응답 시간을 나타낸다. 그림 9의 결과에서 볼 수 있듯이 한 충돌 도메인 내의 노드들을 관리하고자 하는 경우SNMP가 이동에이전트나 혼합모드보다 응답시간이 작음을 알 수

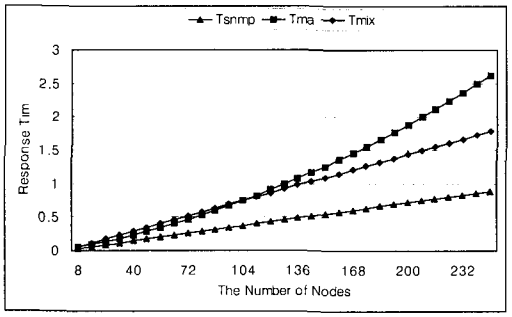


그림 9. 관리 노드 수에 따른 전체 응답시간

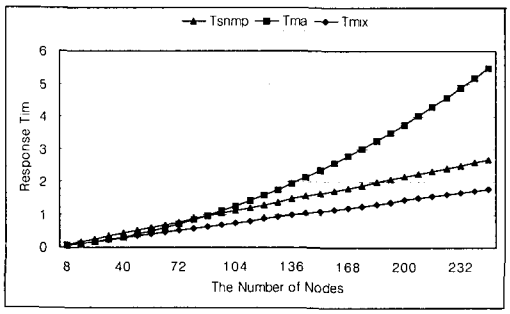


그림 10. 각 노드로 데이터 요구 회수가 3회인 경우의 전체 응답시간

있다. 이는 이동에이전트나 혼합모드의 초기 크기가 크므로 패킷이 전송되는데 어려운 반면 SNMP는 클라이언트-서버 방식에도 불구하고 각 노드로 이동시 한 개의 패킷으로 요구와 응답을 모두 처리할 수 있기 때문이다. SNMP나 혼합모드의 경우 노드 수의 증가에 따라 응답시간이 선형적으로 증가하는 반면 이동에이전트는 데이터가 이동에이전트에 누적되어 관리가 이루어지기 때문에 지수적인 증가의 특성을 보인다. 그림 10은 각 노드로 데이터 요구 회수가 3회인 경우이다. 예를 들어 Interface utilization같은 관리 작업을 하고자 할 때 SNMP는 각 노드 당 NMS와 3회씩 요구와 응답을 받아야 하지만 이동에이전트나 혼합모드는 관리 노드에서 요구와 응답이 이루어지므로 NMS와의 통신이 불필요하다. 그래프 결과에서 보듯이 혼합 모드는 데이터 요구 회수 즉 태스크에 영향을 받지 않으므로 응답시간이 가장 작게 나오고 관리노드의 개수가 약 100개 미만에서는 이동에이전트가 SNMP보다 응답시간이 작은 반면에 노드 수가 증가함에 따라 이동에이전트의 성능이 나빠짐을 알 수 있다. 그림 11은 각 노드로 데이터 요구 회수가 3회인 경우 이동에이전트의 크기 변화에 따른 응답시간을 보여준

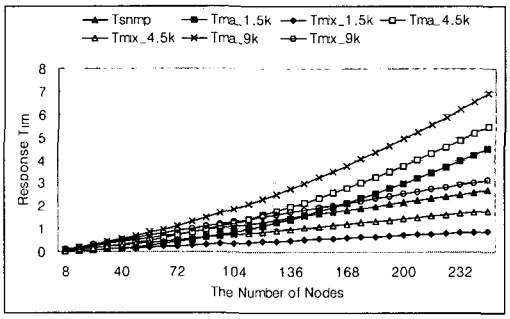


그림 11. 이동에이전트 크기 변화에 따른 응답시간

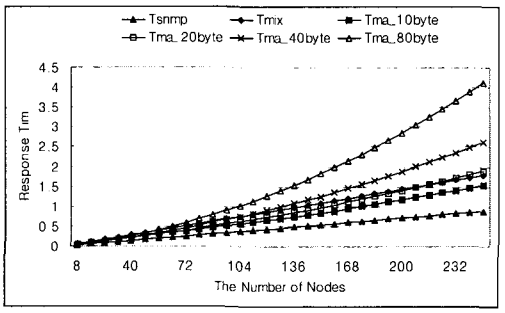


그림 12. 태스크에 따른 응답 시간 비교

다. T_{SNMP}을 기준으로 봤을 때 이동에이전트의 크기가 4.5Kbyte보다 작으면 노드의 개수가 100개 미만에서 더 좋은 응답 시간을 보임을 알 수 있다. 따라서 일반적인 이동에이전트로 네트워크 관리를 설계하기 위해서는 관리 노드의 개수를 100개미만으로 충돌 도메인을 구성하는 것이 좋음을 알 수 있다.

그림 12는 이동에이전트의 크기는 고정시키고 관리 노드에서 얻어오는 MIB_{value}값을 변화시킴에 따라 응답 시간을 비교한 것이다. 결과에서 보듯이 이동에이전트로 관리하고자 하는 경우 관리 노드가

100개가 넘어서면 각 노드에서 획득하는 MIB_{value} 가 증가함에 따라 관리 응답 시간이 빠르게 증가한다. 즉, 이동에이전트는 관리 태스크에 민감한 반응을 보임을 알 수 있다

4.2 WAN환경에서의 실험

본 논문에서 WAN환경이라 함은 인트라넷과 인터넷을 같이 생각한다. 인트라넷은 경로 AS의 지연이 작은 반면에 인터넷은 지연을 크게 함으로써 실험을 한다. 모든 파라미터 값은 LAN환경에서 사용

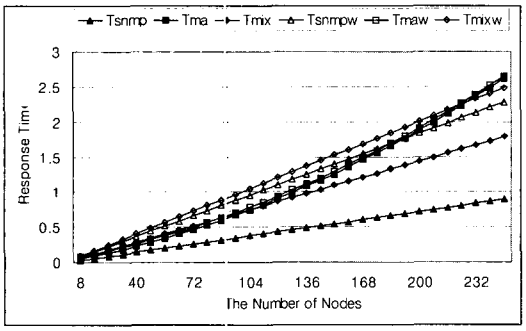


그림 13. LAN vs WAN(인트라넷)환경의 응답시간 비교

했던 것을 그대로 사용한다. 그림 13은 한 충돌 도메인에 NMS와 관리노드들이 같이 있는 LAN환경과 NMS는 관리 도메인에 있고 모든 관리 노드들이 다른 충돌 도메인에 존재할 때의 응답시간을 비교한 결과이다. 인트라넷 환경을 위해 경로 AS의 지연을 1ms로 설정했다. 그래프에서 보듯이 지연이 없는 LAN환경에서는 SNMP를 사용하는 것이 유리하지만 SNMP는 작은 지연 시간을 가지는 인트라넷 환경에서 조차 응답 시간이 급격히 증가함을 알 수 있다. 이에 비해 이동에이전트는 위 그래프에서 약간의 응답 시간만이 증가함을 알 수 있다.

그림 14는 경로 AS의 지연 시간 변화에 따른 전체 응답 시간을 비교한 것이다. 지연을 1ms부터 10ms까지 변화 시키며 모든 관리 노드 248개를 방문한 결과이다. 그래프에서 보듯이 SNMP의 경우 모든 요구와 응답 메시지가 경로 AS을 거쳐서 관리가 이루어지므로 지연에 매우 민감함을 알 수 있다. 반면에 이동에이전트는 지연에 대한 변화가 거의 없으므로 지역적으로 멀리 떨어져 지연이 큰 네트워크를 관리하고자 할 때 유리함을 알 수 있다. 혼합모드의 경우 관리 노드에 저장된 데이터를 TCP/IP를 사용해 NMS로 전송할 때 지연이 발생하는데 SNMP와 이동에이전트의 중간 정도의 응답시간을 갖는다.

그림 15는 네트워크 관리를 위해 관리 노드의 MIB_{value}을 얻기 위해 3회씩 방문해야 하고 지연이 1ms와 8ms인 경우의 결과를 보인 것이다. 지연을 1ms로 실험했을 때 이동에이전트와 혼합모드를 살펴보면 혼합모드가 더 좋은 응답시간을 보임을 알 수 있다. 이는 이동에이전트가 각 관리 노드에서 많은 데이터를 획득하기 때문에 관리 노드 간을 이주시 많은 지연이 발생한 반면 혼합 모드는 관리

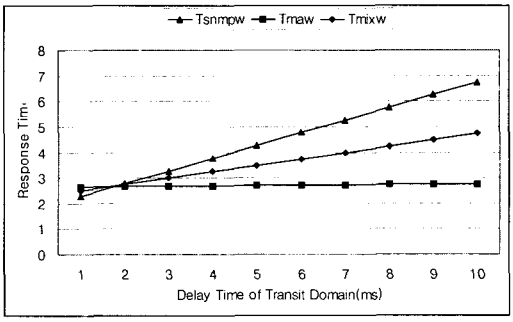


그림 14. 경로 AS의 지연 시간에 따른 응답시간 비교

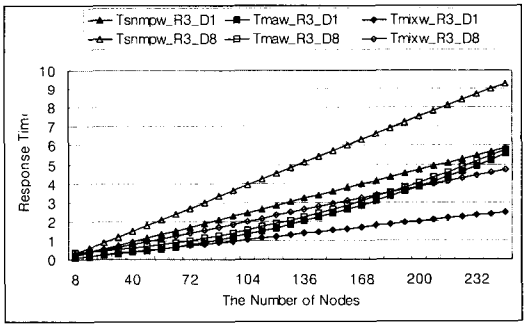


그림 15. 태스크와 지연의 변화에 따른 응답 시간 비교

태스크에 상관없이 일정한 지연 시간을 가지기 때문이다. 지연을 8ms로 실험한 경우 관리 노드가 150개 이하에서는 이동에이전트의 응답시간이 혼합모드보다 좋아짐을 알 수 있다. 즉 이동에이전트의 응답 시간은 관리하고자 하는 태스크에 많은 영향을 받으며 혼합모드의 응답 시간은 이동에이전트보다 지연에 민감하다. SNMP는 모든 경우에 가장 좋지 않은 응답 시간을 보이므로 WAN환경에서 네트워크 관리를 하고자 할 때는 부적절함을 알 수 있다.

4.3 적응형 네트워크 관리 알고리즘

해석적 모델의 분석 결과 클라이언트-서버 방식인 SNMP는 경로 AS의 지연에 매우 민감하고 분산형 방식인 이동에이전트의 응답시간은 관리하고자 하는 태스크에 많은 영향을 받은 반면 혼합모드는 태스크에는 영향을 받지 않고 지연에는 SNMP보다 덜 민감하다는 것을 알 수 있다. 제안한 해석적 모델의 결과를 바탕으로 네트워크 환경, 경로 AS의 지연, 관리 태스크, 및 관리 노드 수를 고려한 네트워크 관리 알고리즘을 제안한다. 그림 16은 본 논문에서 제안한 적응형(adaptive) 네트워크 관리 알고리즘을

```
if LAN environment
  if single MIB variable
    use SNMP;
  else multiple MIB variables
    use Mixed Mode;
else WAN environment
  if transit domain delay less than
 2ms(Intranet)
    if single MIB variable
      if the number of nodes less than 100
        use Mobile Agent;
      else the number of node over than 100
        use SNMP;
    else multiple MIB variables
      use Mixed Mode;
  else transit domain delay over than
 2ms(Internet)
    if single MIB variable
      use Mobile Agent;
    else multiple MIB variables
      if the number of nodes less than 100
        use Mobile Agent;
      else the number of node over than 100
        use Mixed Mode;
```

그림 16. 적응형 네트워크 관리 알고리즘

보여준다. 알고리즘 구성은 관리하고자 하는 네트워크 환경에 우선순위를 두고 경로 AS의 지연에 따른 분류, 관리 노드에서 획득해오는 MIB_{value} 값에 따른 태스크에 의한 분류 후 관리 노드 수에 따라 각 네트워크 관리 패러다임을 적용한 것이다.

그림 17은 알고리즘을 실험하기 위한 통합적 네트워크 환경의 예시이다. 관리 도메인에 NMS와 관리 노드들이 존재하고 충돌 도메인은 인트라넷과 인터넷을 거쳐 3개가 존재하는 경우 파라미터 N과 R을 변화시키셔 실험한다. 각 도메인 당 SNMP, 이동에이전트, 혼합모드, 그리고 적응형 모드 각각의 네트워크 관리 패러다임을 적용시켜서 응답시간을 합산하는 방법으로 실험한다. 그림 18은 각 네트워크 관리 패러다임을 그림 17에 제시한 네트워크 환경에 적용한 결과를 보여준다.

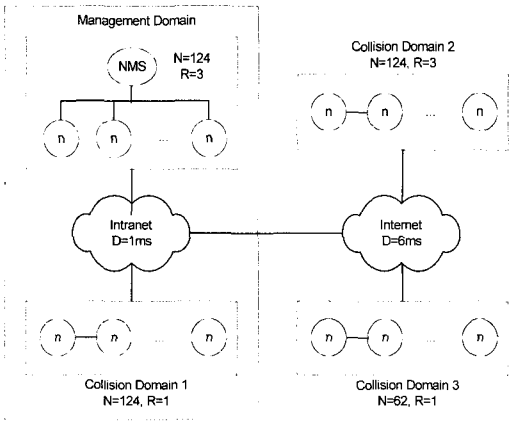


그림 17. 통합적 네트워크 환경

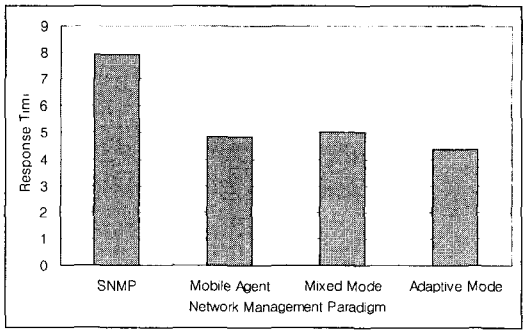


그림 18. 네트워크 관리 패러다임에 따른 응답시간 비교

SNMP는 지연에 매우 민감하기 때문에 분산형 관리 패러다임인 이동에이전트나 혼합모드에 비해 상당히 큰 응답시간을 보인다. 적응형 네트워크 관리 알고리즘을 사용하면 이동에이전트나 혼합모드에 비해 약 10%의 성능 향상이 있음을 알 수 있다.

V. 결 론

본 논문에서는 효율적인 네트워크 관리 시스템을 위해 기존의 클라이언트-서버 패러다임인 SNMP와 최근에 네트워크 관리를 위해 도입되는 분산 패러다임인 이동 에이전트 및 혼합모드에 대해 해석적 모델을 제시한 후 각 파라미터 값의 변화에 따른 네트워크 응답 시간을 비교하였다. 제안한 해석적 모델을 네트워크 응답 시간에 중점을 두어 성능 평가 한 결과 LAN에서는 대체적으로 SNMP가 유리한 반면 WAN에서는 네트워크 환경에 따라 이동에이전트 또는 혼합모드가 더 좋은 응답 시간을 보임을 알 수 있다. 클라이언트-서버 방식인 SNMP는 지연에 매우 민감하고 분산형 방식인 이동에이전트의 응답시간은 관리하고자 하는 태스크에 많은 영향을 받은 반면 혼합모드는 태스크에는 영향을 받지 않고 지연에는 SNMP보다 덜 민감하다는 것을 알 수 있다. 또한 해석적 방법의 결과를 바탕으로 네트워크 환경, 경로 AS의 지연, 관리 태스크, 관리 노드 수를 고려한 적응형 네트워크 관리 알고리즘을 제안한 후 실험한 결과 한 가지 네트워크 관리 패러다임을 사용하는 것보다 네트워크 관리 응답시간을 줄일 수 있었다. 향후 연구 계획으로는 성능 평가를 위해 제안한 해석적 모델을 구현을 통해 검증해 보아야 한다.

참 고 문 헌

[1] W. Stallings, *SNMP, SNMPv2, SNMPv3, and RMON1 and 2* (2nd Edition), Addison-Wesley, 1999.

[2] M. Baldi and G. P. Picco, "Evaluation The Tradeoffs of Mobile Code Design Pradigms in Network Management Applications," ICSE'98, Apr. 1998.

[3] A. Bieszczad, T. White and B. Pagurek, "Mobile Agents for Network Management," IEEE Communication Surveys, Sep. 1998.

[4] A. Sahai and C. Morin, "Towards Distributed and Dynamic Network Management", Proc. of the IEEE/IFIP Network Operation and Management Symposium, Feb. 1998.

[5] G. Susilo, A. Biezczad and B. Pagurek, "Infrastructure for Advanced Network Management Based on Mobile Code," Proc. of NOMS'98, Feb. 1998.

[6] C. G. Harrison, D. M. Chess, A. Kershenbaum "Mobile Agent: Are they a good idea?," IBM Research Division, Mar. 1995.

[7] IETF, *Management Information Base for Network Management of TCP/IP-based Internets: MIB-II*, Mar. 1991.

[8] J. D. Case, K. McCloghrie, M. Rose and S. Waldbusser, "Structure of Management Information for Version 2 of the Simple Network Management Protocol," RFC 1902, Jan. 1996.

[9] S. Waldbusser, "Remote Network Monitoring Management Information Base," RFC 1757, Feb. 1995.

[10] A. Fuggetta, G. P. Picco, G. Vigna, "Understanding Code Mobility," IEEE Transactions on Software Engineering, vol. 24, May. 1998.

[11] H. S. Nwana, "Software Agents: an Overview," Knowledge Engineering Review, vol. 11, Sep. 1996.

[12] Y. Aridor, D. B. Lange, "Agent Design Patterns: Elements of Agent Application Design," Proceedings of Autonomous Agents'98, Sep. 1998.

[13] Mischa Schwartz, *Telecommunication Networks: Protocols, Modeling and Analysis*, Addison-Wesley, 1987.

[14] L. Kleinrock, *Queueing Systems Volume 1: Theory*, John Wiley & Sons, 1975.

[15] C. Pattinson, "A Study of the Behaviour of the Simple Network Management Protocol," DSOM'2001, Oct. 2001.

[16] E. W. Zegura, K. L. Calvert and M. J. Donahoo, "A quantitative comparison of graph-based models for internet topology," IEEE/ACM Transaction on Networking, Dec. 1997.

이 정 우(Jung-Woo Lee) 학생 회원

1992년 2월 : 인하대학교 전자 공학과 졸업
2003년 2월 : 인하대학교 전자 공학과 석사

<주관심분야> 컴퓨터 네트워크, 네트워크 매니지먼트 시스템

윤 완 오(Wan-Oh Yoon) 학생 회원

2000년 2월 : 경기대학교 전자 공학과 졸업
2002년 2월 : 인하대학교 전자 공학과 석사
2002년 3월~현재 : 인하대학교 전자공학과 박사과정

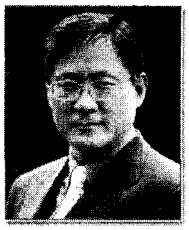
<주관심분야> 분산 처리 시스템, 병렬프로그래밍, 컴퓨터 아키텍처

신 광 식(Kwang-Sik Shin) 학생 회원

2001년 2월 : 인하대학교 전자 공학과 졸업
2003년 2월 : 인하대학교 전자 공학과 석사
2003년 3월~현재 : 인하대학교 전자공학과 박사과정

<주관심분야> 멀티미디어 데이터 통신, 컴퓨터 네트워크, 시스템 프로그램밍

최 상 방(Sang-Bang Choi) 정 회원



1981년 2월 : 한양 대학교 전자공학과 졸업
1981년~1986년 : LG 정보통신(주)
1988년 3월 : University of Washington 석사
1990년 8월 : University of Washington 박사
1991년~현재 : 인하대학교 전자공학과 교수

<주관심분야> 컴퓨터 구조, 컴퓨터 네트워크, 병렬 및 분산 처리 시스템, Fault-tolerant computing